SEARCH-E1: SELF-DISTILLATION DRIVES SELF-EVOLUTION IN SEARCH-AUGMENTED REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Outcome based reinforcement learning provides a simple and effective recipe for training search augmented reasoning agents. Recent work seeks further improvements through stronger external teachers, auxiliary process supervision, specialized reward shaping, and increasingly elaborate training pipelines. This raises a natural question: *can agents continue improving by learning from their own successful experience?* We present **Search-E1**, a simple self evolution framework that alternates vanilla GRPO with on policy self distillation (OPSD). After each GRPO round, Search-E1 mines sibling trajectories generated by the policy itself and exposes the search skeleton of an efficient successful trajectory, that is, the sequence of queries it issued, as privileged context. A token level forward KL objective then aligns the policy under its standard inference context with its own distribution under this privileged context. In this way, the policy converts contrasts among its own rollouts into dense token level supervision without an external teacher, auxiliary reward model, or additional annotation. On seven single hop and multi hop QA benchmarks, Search-E1 reaches an average Exact Match of 0.440 with Qwen2.5-3B and 0.482 with Qwen2.5-7B, achieving the best average performance among all compared methods at both scales. These results show that alternating outcome based exploration with self distillation provides a simple and effective path to iterative improvement in search augmented reasoning.

1 INTRODUCTION

Large language models have demonstrated strong reasoning capabilities across a wide range of tasks (Guo et al., 2025; OpenAI, 2024; Yang et al., 2025), but their parametric knowledge can be incomplete or outdated. Search augmented reasoning addresses this limitation by allowing a model to interleave reasoning with calls to an external retriever, deciding when to search and what to query for (Jin et al., 2025a; Chen et al., 2025; Shi et al., 2025; Yao et al., 2022). Recent work has shown that such agents can be trained effectively with outcome based reinforcement learning such as GRPO (Shao et al., 2024), using final answer correctness as the primary supervision signal. The resulting recipe is simple: sample trajectories, evaluate their outcomes, and optimize the policy with GRPO.

Recent methods seek further improvements by adding increasingly specialized components, including stronger external supervision, process level reward signals, structured rollout strategies, and manually designed reward shaping (Xu et al., 2025; Wang et al., 2025; Zhang et al., 2025a; Feng et al., 2025; Ding & Ye, 2025; Xia et al., 2026). These approaches improve search agents from different perspectives, but also introduce additional models, annotations, objectives, or assumptions into the training pipeline. This raises a natural question: *are these additional components necessary for continued improvement in search augmented reasoning?*

Our starting point is a simple observation. For the same training question, standard GRPO naturally produces sibling trajectories with substantially different behaviors. One trajectory may reach the correct answer through a concise and effective search process, while another may issue redundant queries, follow an unproductive reasoning path, or fail entirely. These contrasts already provide useful supervision for distinguishing more effective behaviors from less effective ones. Rather than introducing an external supervisor to extract this information, we ask whether the policy can directly learn from the successful experience contained in its own rollouts.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

We therefore propose **Search-E1**, a self evolution framework that alternates GRPO with on policy self distillation (OPSD). The two stages serve complementary roles. GRPO explores at the trajectory level and discovers successful behaviors, while OPSD consolidates these discoveries into the policy. After each GRPO round, Search-E1 mines an efficient successful sibling trajectory and exposes only the queries it issued as privileged context, withholding the retrieved evidence and the final answer. The same policy then serves as both student and teacher under different conditioning contexts, and a token level forward KL objective transfers the behavior induced by the privileged context back to the standard inference setting. The resulting training signal is derived entirely from the policy's own rollouts, without an external teacher, auxiliary reward model, or intermediate annotation.

The exploration and consolidation stages can be iterated for multiple rounds. Each GRPO round refreshes the on policy trajectory pool, from which successful trajectories are mined to provide privileged supervision for the subsequent OPSD round. This forms a simple self-evolution loop in which the policy repeatedly learns from experience generated by its own updated behavior. We further study how the effectiveness of this loop changes across rounds in Section 3.3.

On seven single hop and multi hop QA benchmarks, Search-E1 reaches an average Exact Match of 0.440 with Qwen2.5-3B and 0.482 with Qwen2.5-7B, achieving the best average performance among all compared methods at both scales. The gains are particularly pronounced on multi hop benchmarks, where successful trajectories contain richer contrasts across multiple search decisions. These results suggest that search agents can be improved iteratively through a simple loop of outcome based reinforcement learning and self distillation, without relying on increasingly elaborate external supervisory machinery. Our contributions are summarized as follows:

- We propose **Search-E1**, a self evolution framework that alternates vanilla GRPO with on policy self distillation, requiring no external teacher, auxiliary reward model, or intermediate annotation.
- We show how differences among sibling trajectories generated by the policy itself can be converted into dense token level supervision, allowing successful behaviors discovered by reinforcement learning to be consolidated and reused in subsequent rounds.
- Experiments on seven QA benchmarks at the 3B and 7B scales show consistent improvements over strong outcome reward and process supervision baselines. We further analyze the training dynamics that affect the effectiveness of repeated self evolution across rounds.

2 METHOD

Search-E1 alternates two standard learning mechanisms with complementary roles. GRPO explores new behaviors using outcome rewards, while on policy self distillation consolidates successful behaviors discovered by the policy itself. We first introduce the search augmented trajectory format and GRPO objective in Section 2.1. We then describe how self generated successful trajectories provide privileged supervision for OPSD in Section 2.2. Finally, Section 2.3 presents the iterative self evolution procedure. An overview of the complete framework is shown in Figure 1.

2.1 PRELIMINARIES

Search augmented trajectories. Following prior work on search augmented reasoning (Jin et al., 2025a; Shi et al., 2025), a policy π_θ generates a trajectory τ by interleaving reasoning with calls to a fixed retriever $\mathcal{E}$. A trajectory contains `<think>` spans for internal reasoning, `<search>` spans for retrieval queries, `<information>` spans returned by the retriever, and a final `<answer>` span. We denote by $\mathcal{A}_\tau$ the positions generated by the policy itself. Tokens inside `<information>` spans are excluded from $\mathcal{A}_\tau$ because they are produced by the retriever.

Outcome based GRPO. For each question q, GRPO samples a group of G trajectories $\{\tau_1, \ldots, \tau_G\}$ from the rollout policy $\pi_{\theta_{\text{old}}}$. Each trajectory receives an outcome reward R_i according to the correctness of its final answer. The group normalized advantage is

$$\hat{A}_i = \frac{R_i - \text{mean}(\{R_j\}_{j=1}^G)}{\text{std}(\{R_j\}_{j=1}^G)}. \tag{1}$$

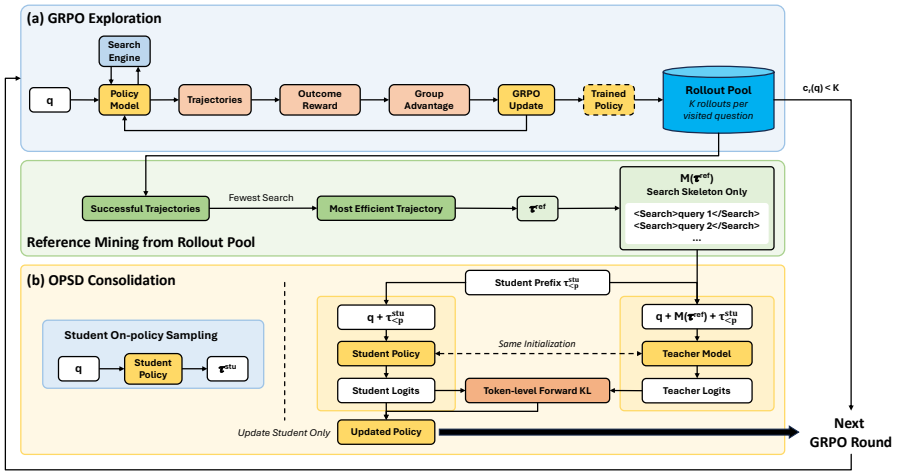

Figure 1: Overview of Search-E1. GRPO explores with outcome rewards and produces an on-policy rollout pool. Successful trajectories provide privileged supervision for OPSD, which consolidates discovered behaviors through token-level self-distillation before the updated policy enters the next GRPO round.

The policy is optimized using the standard GRPO objective

$$\mathcal{L}_{\mathrm{GRPO}}(\theta) = -\mathbb{E}\left[\min\left(r_p(\theta)\hat{A}_i, \mathrm{clip}\left(r_p(\theta), 1-\epsilon, 1+\epsilon\right)\hat{A}_i\right)\right], \tag{2}$$

where

$$r_p(\theta) = \pi_\theta(a_p \mid \tau_{<p}) / \pi_{\theta_{\mathrm{old}}}(a_p \mid \tau_{<p}). \tag{3}$$

Search-E1 leaves this stage unchanged. In particular, we introduce no process reward, auxiliary critic, or modification to the GRPO advantage.

2.2 Self Distillation from Self Generated Privileged Trajectories

The key observation behind Search-E1 is that GRPO already generates the supervision required for further improvement. For the same question, some trajectories solve the problem through an effective search process while others fail or take less efficient paths. We use the successful trajectories generated by the policy itself as privileged information for on policy self distillation.

Mining privileged trajectories. After GRPO round r, we sample K trajectories $\{\tau_q^{(1)}, \ldots, \tau_q^{(K)}\}$ for each question visited during that round using the resulting policy π_r^{RL}. Each trajectory is labeled by its final answer correctness. Among the successful trajectories, we select an efficient trajectory as the privileged reference τ_q^{ref}. By default, we prefer the trajectory with the fewest search calls and break ties by the total length of its search skeleton. Questions for which no successful trajectory is observed do not provide a privileged reference for the current OPSD round. Importantly, τ_q^{ref} is generated entirely by the current policy. It is neither a human annotated reasoning trace nor a trajectory synthesized by a stronger model.

Search skeleton as privileged context. The privileged reference enters the teacher context only through its search skeleton. We define a masking operator $\mathcal{M}$ that retains the `<search>` spans of a trajectory in their original order and discards all remaining content, including `<think>` spans, retrieved `<information>` passages, and the final `<answer>` span. The teacher therefore observes which queries were issued and in what order, but neither the evidence they returned nor the answer they supported. This restriction is deliberate. Retaining the retrieved passages or the final answer would make the answer directly predictable from the teacher context, so the teacher would place mass on continuations that skip retrieval and would push the student toward answering without evidence. The skeleton also keeps the privileged context short, since retrieved passages dominate the length of a search augmented trajectory. We isolate the effect of this choice in Section 3.5 and analyze it further in Appendix D.

On policy student sampling. Given a question q with privileged reference τ_q^{ref}, the student samples its own trajectory

$$\tau^{\mathrm{stu}} \sim \pi_\theta(\cdot \mid P_S(q)), \tag{4}$$

where $P_S(q)$ is the standard inference prompt containing the task instruction and question. Thus, the distillation objective is evaluated on trajectories generated by the policy being optimized, rather than on the privileged reference itself.

Privileged teacher conditioning. The student and teacher originate from the same policy but receive different information. The student observes only the standard inference context,

$$P_S(q) = [\text{instruction}] \,\|\, q, \tag{5}$$

while the teacher additionally observes the search skeleton of the self generated privileged reference,

$$P_T(q, \tau_q^{\mathrm{ref}}) = [\text{instruction}] \,\|\, q \,\|\, \mathcal{M}(\tau_q^{\mathrm{ref}}). \tag{6}$$

Both policies evaluate the same student generated trajectory τ^{stu}. For each policy generated position $p \in \mathcal{A}_{\tau^{\mathrm{stu}}}$, we define

$$P_p^{\mathrm{stu}} = \pi_\theta\left(\cdot \mid P_S(q), \tau_{<p}^{\mathrm{stu}}\right), \tag{7}$$

$$P_p^{\mathrm{tch}} = \mathrm{sg}\left[\pi_r^{\mathrm{RL}}\left(\cdot \mid P_T(q, \tau_q^{\mathrm{ref}}), \tau_{<p}^{\mathrm{stu}}\right)\right], \tag{8}$$

where $\mathrm{sg}[\cdot]$ denotes stop gradient. At the beginning of each OPSD round, both teacher and student are initialized from π_r^{RL}. The teacher is held fixed during distillation, while the student is updated. The distinction between the two therefore arises from privileged conditioning rather than from a stronger external model.

Token level distillation. Following OPSD (Zhao et al., 2026), we minimize a forward KL divergence between teacher and student along the student's own rollout. We use pointwise clipping to prevent a small number of large vocabulary level divergences from dominating the update:

$$\mathcal{L}_{\mathrm{OPSD}} = \frac{1}{|\mathcal{A}_{\tau^{\mathrm{stu}}}|} \sum_{p \in \mathcal{A}_{\tau^{\mathrm{stu}}}} \sum_{v \in \mathcal{V}} \min\left(P_p^{\mathrm{tch}}(v) \log \frac{P_p^{\mathrm{tch}}(v)}{P_p^{\mathrm{stu}}(v)}, \tau_{\mathrm{clip}}\right). \tag{9}$$

The teacher never generates the student response. It only provides a privileged next token distribution over the student's on policy trajectory. Consequently, the student receives dense token level supervision while remaining on its own generation distribution. At inference time, the privileged context and the teacher evaluation are removed.

2.3 Iterative Self Evolution

Search-E1 alternates GRPO exploration and OPSD consolidation over multiple rounds. Suppose round r starts from policy π_r. GRPO first optimizes π_r using only outcome rewards, producing π_r^{RL}. We then sample a rollout pool from π_r^{RL}, mine successful privileged trajectories, and apply OPSD to obtain the policy π_{r+1}. The resulting policy enters the next GRPO round. The rollout pool also determines which questions remain useful for subsequent exploration. Let $\mathcal{V}_r \subseteq \mathcal{D}_r$ denote the questions visited during round r, and let $c_r(q)$ denote the number of correct trajectories among the K rollouts sampled from π_r^{RL} for a question $q \in \mathcal{V}_r$.

For OPSD, we use questions for which at least one successful trajectory is available:

$$\mathcal{D}_r^{\mathrm{OPSD}} = \{q \in \mathcal{V}_r : c_r(q) > 0\}. \tag{10}$$

For the next GRPO round, we retain questions that are not uniformly solved:

$$\mathcal{D}_{r+1} = \mathcal{D}_r \setminus \{q \in \mathcal{V}_r : c_r(q) = K\}. \tag{11}$$

All failure questions remain for the next GRPO round but provide no OPSD reference; mixed outcome questions participate in both stages; uniformly solved questions may provide OPSD supervision in the current round but are removed from the next GRPO round.

The resulting procedure forms a closed self evolution loop. GRPO discovers successful behaviors through exploration, the policy's own rollout pool supplies privileged supervision, and OPSD consolidates these behaviors before the next round begins. No external teacher, process reward model, or additional annotation is introduced at any stage.

3 EXPERIMENTS

We evaluate Search-E1 with four questions. First, does the simple alternation of GRPO and OPSD improve over existing search augmented reasoning methods? Second, do the gains accumulate across multiple self evolution rounds? Third, what determines whether an OPSD updated policy remains suitable for further GRPO exploration? Finally, which components of trajectory mining and data recycling are necessary for the observed improvements?

3.1 EXPERIMENTAL SETUP

Datasets and evaluation. Following the evaluation protocol of prior search augmented reasoning work (Jin et al., 2025a; Shi et al., 2025), we train on the combined training sets of Natural Questions (NQ) (Kwiatkowski et al., 2019) and HotpotQA (Yang et al., 2018), containing approximately 170K question answer pairs. We evaluate on seven QA benchmarks, including three single hop datasets, NQ, TriviaQA (Joshi et al., 2017), and PopQA (Mallen et al., 2023), and four multi hop datasets, HotpotQA, 2WikiMultihopQA (2Wiki) (Ho et al., 2020), MuSiQue (Trivedi et al., 2022), and Bamboogle (Press et al., 2023). Performance is measured by Exact Match (EM) after standard answer normalization, including lowercasing, punctuation removal, and article removal. Detailed dataset statistics are provided in Appendix B.

Baselines. We compare Search-E1 with retrieval free baselines, single retrieval methods, and multi round search augmented reasoning methods. Retrieval free baselines include Direct Generation, supervised fine tuning, and R1 style reinforcement learning (Guo et al., 2025). We include Naive RAG (Lewis et al., 2021) as a single retrieval baseline and Search-o1 (Li et al., 2025) and IR-CoT (Trivedi et al., 2023) as inference time search methods. For reinforcement learning based search agents, we compare against Search-R1 (Jin et al., 2025a), ReSearch (Chen et al., 2025), AutoRefine (Shi et al., 2025), and MR-Search (Xiao et al., 2026). We further include methods that introduce denser supervision or additional training structure, including StepSearch (Wang et al., 2025), GiGPO (Feng et al., 2025), Thinker (Xu et al., 2025), SD-Search (Ma et al., 2026), and IG-Search (Liang et al., 2026). Baseline numbers are taken from the original papers when the evaluation protocol is compatible.

Models and retrieval. We evaluate both Base and Instruct variants of Qwen2.5 at the 3B and 7B scales (Qwen, 2025). All methods use the same fixed retrieval environment whenever applicable. The retriever is E5-base-v2 (Wang et al., 2024) over a December 2018 Wikipedia corpus and returns the top three passages for each query. Each rollout is allowed at most four search calls. We use a rollout temperature of 1.0 throughout training and evaluation.

Training configuration. Search-E1 is implemented with veRL (Sheng et al., 2025). GRPO uses an Exact Match outcome reward with group size $G = 5$, learning rate 1×10^{-6}, KL coefficient $\beta = 0.001$, and clip ratio $\epsilon = 0.2$. After each GRPO round, we sample $K = 5$ trajectories for each question visited during that round to construct the rollout pool used for privileged trajectory mining and data recycling, so mining and the recycling rules of Equations 10 and 11 are applied to the questions actually visited in round r. OPSD uses forward KL with pointwise clipping at $\tau_{\text{clip}} = 10$. The privileged context contains only the search skeleton of the reference trajectory and is truncated to 512 tokens. The distillation loss is applied at all policy generated positions of the student rollout. For the transition to the next GRPO round, we select the OPSD checkpoint with the highest validation EM among checkpoints whose mean token entropy retains at least 85% of its value at the beginning of the OPSD round. We run three GRPO and OPSD cycles for the 3B models and two cycles for the 7B models. Unless otherwise specified, ablations are conducted with Qwen2.5-3B-Instruct. Complete hyperparameters are reported in Appendix C.

3.2 MAIN RESULTS

Table 1 reports results at the 3B scale. At the 3B scale, Search-E1-Instruct achieves the highest average EM of 0.440, while Search-E1-Base reaches 0.438. Both variants outperform the strongest outcome reward baseline, MR-Search-Base (0.414), by more than two points on average. They also improve over the strongest variants of SD-Search and IG-Search, which reach average EM scores

Table 1: Exact Match on seven QA benchmarks with Qwen2.5-3B. **Bold** denotes the best result and underline the second best in each column. "–" denotes results not reported by the original paper.

	Single-Hop			*Multi-Hop*				
Method	NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	**Avg.**
Outcome Reward and Standard Search Training								
Search-R1-Base	0.421	0.583	0.413	0.297	0.274	0.066	0.128	0.312
Search-R1-Instruct	0.397	0.565	0.391	0.331	0.310	0.124	0.232	0.336
ReSearch-Base	0.427	0.597	0.430	0.305	0.272	0.074	0.128	0.319
ReSearch-Instruct	0.365	0.571	0.395	0.351	0.272	0.095	0.266	0.331
AutoRefine-Base	0.467	0.620	0.450	0.405	0.393	0.157	0.344	0.405
AutoRefine-Instruct	0.436	0.597	0.447	0.404	0.380	0.169	0.336	0.396
MR-Search-Base	0.477	0.635	0.460	0.419	0.401	0.165	0.344	0.414
Additional Process or Step Level Supervision								
StepSearch-Base	–	–	–	0.329	0.339	0.181	0.328	–
StepSearch-Instruct	–	–	–	0.345	0.320	0.174	0.344	–
GiGPO-Instruct	0.420	0.595	0.424	0.369	0.370	0.126	**0.641**	0.421
Thinker-Instruct	0.439	0.598	0.469	0.400	**0.469**	**0.214**	0.424	0.430
SD-Search-Base	0.470	0.624	0.467	0.425	0.420	0.188	0.402	0.428
SD-Search-Instruct	0.469	0.624	0.465	0.424	0.419	0.184	0.404	0.427
IG-Search-Base	0.476	0.637	0.476	**0.436**	0.415	0.179	0.390	0.430
IG-Search-Instruct	0.450	0.614	0.460	0.431	0.440	0.191	0.424	0.430
Search-E1-Base (Ours)	**0.479**	**0.641**	**0.478**	0.432	0.429	0.190	0.416	0.438
Search-E1-Instruct (Ours)	0.474	0.626	0.461	0.427	0.436	0.193	0.464	**0.440**

of 0.428 and 0.430, respectively. These results indicate that repeated alternation between standard outcome reward RL and self distillation can outperform methods that introduce additional step level objectives, despite leaving the underlying GRPO objective unchanged.

The two Search-E1 variants show complementary behavior. The Base model performs better on the single hop benchmarks, whereas the Instruct model is stronger on several multi hop datasets, including 2Wiki, MuSiQue, and Bamboogle. The improvements over outcome reward baselines are also larger on the multi hop side, where different rollouts of the same question exhibit greater variation in their intermediate search and reasoning behavior. This provides more informative privileged trajectories for subsequent self distillation.

Table 2 reports the corresponding results at the 7B scale. At the 7B scale, Search-E1 shows the same overall pattern. Search-E1-Instruct reaches an average EM of 0.482, while the Base variant reaches 0.480. Both outperform the strongest standard outcome reward baseline, MR-Search-Base (0.460), as well as the strongest compared method with additional step level supervision, IG-Search-Instruct (0.479). Both Search-E1 variants improve over their corresponding SD-Search and IG-Search counterparts. Search-E1-Base reaches 0.480, compared with 0.471 for SD-Search-Base and 0.476 for IG-Search-Base, while Search-E1-Instruct reaches 0.482, compared with 0.476 and 0.479, respectively. The relative strengths of the Base and Instruct variants remain consistent with the 3B results. The Base model performs particularly well on the single hop benchmarks, whereas the Instruct model attains higher performance on 2Wiki and MuSiQue. Importantly, the improvement of Search-E1 persists as the backbone grows from 3B to 7B, suggesting that the benefit of self generated privileged supervision is not restricted to smaller policies.

3.3 Self Evolution Across Rounds

The final results establish the overall effectiveness of Search-E1, but do not reveal how the improvement accumulates across successive rounds. We therefore evaluate the policy after every GRPO and OPSD stage. Figure 2 reports the average EM throughout the complete self evolution process.

At the 3B scale, both Base and Instruct models improve consistently over three cycles. Search-E1-Base increases from 0.413 after the first GRPO round to 0.438 after the third OPSD round, while Search-E1-Instruct improves from 0.415 to 0.440. Importantly, the gains are not confined to

Table 2: Exact Match on seven QA benchmarks with Qwen2.5-7B. **Bold** denotes the best result and underline the second best in each column. "–" denotes results not reported by the original paper.

	Single-Hop			*Multi-Hop*				
Method	NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	**Avg.**
Outcome Reward and Standard Search Training								
Search-R1-Base	0.469	0.627	0.449	0.410	0.272	0.173	0.456	0.408
Search-R1-Instruct	0.393	0.610	0.397	0.370	0.414	0.146	0.368	0.385
AutoRefine-Base	0.484	0.659	0.487	0.451	0.405	0.187	0.512	0.455
AutoRefine-Instruct	0.473	0.652	0.475	0.443	0.398	0.181	0.496	0.445
MR-Search-Base	0.502	0.666	0.472	0.468	0.436	0.221	0.452	0.460
Additional Process or Step Level Supervision								
StepSearch-Base	–	–	–	0.380	0.385	0.216	0.467	–
StepSearch-Instruct	–	–	–	0.386	0.366	0.226	0.400	–
GiGPO-Instruct	0.464	0.647	0.461	0.416	0.436	0.189	**0.689**	0.472
Thinker-Instruct	0.450	0.642	0.484	0.421	**0.469**	0.221	0.480	0.452
SD-Search-Base	0.500	0.665	0.489	0.468	0.438	0.204	0.532	0.471
SD-Search-Instruct	0.495	0.668	0.494	0.471	0.441	0.222	0.544	0.476
IG-Search-Base	0.501	0.675	0.504	0.473	0.430	0.205	0.544	0.476
IG-Search-Instruct	0.490	0.665	0.495	**0.480**	0.445	0.215	0.560	0.479
Search-E1-Base (Ours)	**0.503**	0.683	**0.505**	0.472	0.447	0.225	0.525	0.480
Search-E1-Instruct (Ours)	0.487	**0.694**	0.497	0.469	0.455	**0.236**	0.534	**0.482**

the distillation stages. Each OPSD round improves upon the preceding GRPO checkpoint, and the subsequent GRPO round is able to further improve from the distilled policy. For example, Search-E1-Instruct improves from 0.415 to 0.421 after the first OPSD stage and then reaches 0.429 after the following GRPO round. This alternating improvement indicates that OPSD not only consolidates successful behaviors discovered by GRPO, but also preserves sufficient room for subsequent outcome based exploration.

The gains gradually diminish in later cycles. For Search-E1-Instruct, the improvement from the first GRPO checkpoint to the first OPSD checkpoint is 0.006, whereas the final OPSD stage adds 0.003. A similar trend is observed for the Base model. This suggests that repeated self evolution approaches saturation rather than producing unstable or unbounded improvement.

The same pattern persists at the 7B scale. Search-E1-Base improves from 0.451 after the first GRPO round to 0.480 after two complete cycles, while Search-E1-Instruct improves from 0.449 to 0.482. These results show that alternating outcome based exploration with self distillation provides cumulative gains across rounds at both model scales. Per benchmark results for every stage of all four backbones are reported in Appendix F.

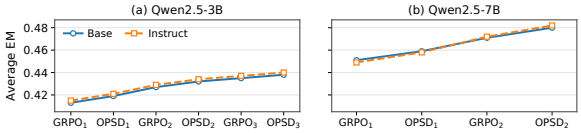

Figure 2: Stage wise average Exact Match during Search-E1 training. Each cycle alternates GRPO exploration with OPSD consolidation. Performance improves after both stages across successive cycles, while the magnitude of improvement gradually decreases in later rounds.

3.4 Preserving Evolvability Across Rounds

The previous results show that GRPO and OPSD can produce cumulative improvements across multiple cycles. A natural question is how to choose the OPSD checkpoint that initializes the next GRPO round. Selecting the checkpoint with the highest immediate validation accuracy appears

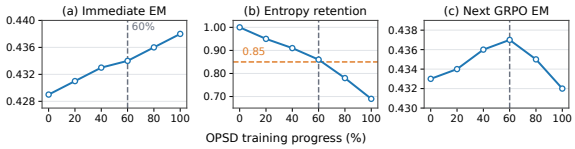

Figure 3: Effect of OPSD checkpoint selection on Qwen2.5-3B-Instruct. As OPSD progresses, immediate validation EM continues to improve, while policy entropy steadily decreases. However, initializing the next GRPO round from increasingly late OPSD checkpoints reduces downstream performance. The vertical dashed line marks the checkpoint selected by our entropy aware handoff rule, while the horizontal dashed line in panel (b) denotes the 0.85 entropy retention threshold.

reasonable, but this criterion ignores whether the resulting policy remains sufficiently diverse for subsequent exploration.

We study this effect by evaluating multiple checkpoints from the second OPSD round of Qwen2.5-3B-Instruct. For each checkpoint, we measure its immediate validation EM, its mean token entropy over policy generated positions on a fixed calibration set relative to the beginning of the OPSD round, and the final EM obtained after using it to initialize an otherwise identical GRPO round. Figure 3 summarizes the results.

Immediate validation performance improves steadily as OPSD progresses, increasing from 0.429 before distillation to 0.438 at the final checkpoint. This improvement is accompanied by a substantial reduction in policy entropy. The entropy retention ratio decreases from 1.00 to 0.69, indicating that extended distillation progressively sharpens the policy distribution.

This sharpening improves current performance but eventually becomes unfavorable for further reinforcement learning. The subsequent GRPO round reaches its highest EM of 0.437 when initialized from an intermediate OPSD checkpoint with an entropy retention of 0.86. Continuing OPSD increases immediate EM to 0.438, but reduces the downstream GRPO result to 0.432. Thus, the checkpoint that performs best immediately after distillation is not necessarily the checkpoint from which the policy can improve most in the next round.

This observation motivates the entropy aware handoff rule used in Search-E1. Among OPSD checkpoints that retain at least 85% of the entropy at the beginning of the round, we select the checkpoint with the highest validation EM. The rule preserves the performance benefit of self distillation while avoiding excessive distribution sharpening that can limit subsequent exploration. Additional analysis in Appendix K shows that the entropy reduction is accompanied by lower rollout diversity, including fewer distinct trajectories and smaller pairwise trajectory differences. Appendix L sweeps the retention threshold and Appendix N verifies that the gain does not come from a change in retrieval.

3.5 Ablation Studies

Table 3 isolates the main design choices of Search-E1. Unless otherwise specified, all variants use the same three cycle schedule, the same question pool, and Qwen2.5-3B-Instruct. Appendix G compares further alternatives to the consolidation stage, and Appendix J varies the divergence and the alignment scope.

Is alternating self distillation necessary? Removing OPSD and running GRPO alone under the same three round schedule reduces the average EM from 0.440 to 0.426. This shows that the improvement across rounds cannot be explained simply by allocating additional outcome reward optimization. The effect is particularly pronounced on the multi hop benchmarks, where the average decreases from 0.380 to 0.365. Since Search-E1 executes more optimizer steps in total than this variant, Appendix E further reports a control that runs GRPO alone for the full number of steps executed by Search-E1, which saturates at 0.427 and then declines.

The form of consolidation also matters. Replacing OPSD with supervised fine tuning on the same mined trajectories, which is the most direct way of reusing self generated experience, reaches only 0.428. Imitating the reference sequence recovers a small part of the benefit, whereas aligning the student to the teacher distribution along its own rollout supplies supervision at every position the

Table 3: Ablation study on Qwen2.5-3B-Instruct. Single and Multi denote the average Exact Match over the three single hop and four multi hop benchmarks, respectively.

Configuration	Single	Multi	Avg.
Search-E1	**0.520**	**0.380**	**0.440**
Iterative Training			
GRPO only	0.507	0.365	0.426
SFT on mined trajectories	0.512	0.365	0.428
Distillation once after all GRPO rounds	0.514	0.370	0.432
Reuse first round reference pool	0.511	0.369	0.430
w/o entropy aware handoff	0.513	0.375	0.434
Trajectory and Data Selection			
Full trajectory as privileged context	0.506	0.356	0.420
Random successful reference	0.516	0.374	0.435
Longest successful reference	0.510	0.372	0.431
w/o question recycling	0.510	0.374	0.432

student actually visits. Applying a single distillation stage after three GRPO rounds, rather than alternating, reaches 0.432. Consolidation is therefore not simply a post training step. Its value depends on being interleaved with exploration, so that every round of reinforcement learning starts from a policy that has already absorbed the behaviors discovered in the previous one.

How should self generated experience be used? What the teacher is allowed to see matters more than any other choice we examined. Passing the full reference trajectory, including its retrieved passages and final answer, reduces the average EM to 0.420, which is below the 0.426 obtained without any distillation at all. A teacher that can already read the answer from its own context assigns low probability to issuing further queries, so the student is pushed toward answering on weaker evidence. Appendix D shows that this variant also reduces the number of search calls per rollout from 2.12 to 1.38 and lowers the fraction of retrievals that return the gold answer. Restricting the privileged context to the search skeleton removes this failure mode. Appendix P traces a single question through the schedule to show what this change looks like on one rollout.

Among skeletons, which successful trajectory is selected still matters. Replacing the efficient successful trajectory with a randomly selected successful trajectory reduces average EM to 0.435, while selecting the longest successful trajectory further decreases performance to 0.431. Successful trajectories therefore do not provide equally useful supervision. Favoring solutions that reach the correct answer with fewer search calls provides a cleaner privileged context for distillation.

Finally, retaining every visited question for the next exploration round, rather than removing those that all rollouts solve, reduces performance from 0.440 to 0.432. A question that every rollout answers correctly yields a zero normalized advantage, so it consumes rollout budget without contributing gradient, and the share of such questions grows with every round. Removing them concentrates later exploration on the failure and mixed outcome questions, where search behavior can still change. Appendix H compares alternative recycling rules and Appendix I compares alternative reference selection rules.

4 Conclusion

We present Search-E1, a simple self evolution framework for search augmented reasoning that alternates vanilla GRPO with on policy self distillation. By using successful trajectories generated by the policy itself as privileged supervision, Search-E1 enables iterative improvement without external teachers, process reward models, or additional annotations. Across seven QA benchmarks, Search-E1 achieves average EM scores of 0.440 at the 3B scale and 0.482 at the 7B scale, outperforming all compared methods. Our results suggest that alternating outcome based exploration with self distillation provides a simple and effective path toward self evolving search agents.

AI USE STATEMENT

No AI tools were used at any stage of this work, including research design, data collection and analysis, code development, experimentation, or manuscript drafting, revision, and language polishing.

REFERENCES

Rishabh Agarwal, Nino Vieillard, Yongchao Zhou, Piotr Stanczyk, Sabela Ramos Garea, Matthieu Geist, and Olivier Bachem. On-policy distillation of language models: Learning from self-generated mistakes. In *The Twelfth International Conference on Learning Representations*, 2024.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. Improving language models by retrieving from trillions of tokens. *arXiv preprint arXiv:2112.04426*, 2022.

Mingyang Chen, Tianpeng Li, Haoze Sun, Yijie Zhou, Chenzheng Zhu, Haofen Wang, Jeff Z. Pan, Wen Zhang, Huajun Chen, Fan Yang, Zenan Zhou, and Weipeng Chen. Research: Learning to reason with search for llms via reinforcement learning. *arXiv preprint arXiv:2503.19470*, 2025.

Zheng Ding and Weirui Ye. Treegrpo: Tree-advantage grpo for online rl post-training of diffusion models. *arXiv preprint arXiv:2512.08153*, 2025.

Lang Feng, Zhenghai Xue, Tingcong Liu, and Bo An. Group-in-group policy optimization for LLM agent training. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, and Wu. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638, 2025.

Chuzhan Hao, Wenfeng Feng, Yuewei Zhang, and Hao Wang. Dynasearcher: Dynamic knowledge graph augmented search agent via multi-reward reinforcement learning. *arXiv preprint arXiv:2507.17365*, 2025.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. Constructing a multi-hop QA dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pp. 6609–6625, 2020.

Zhenyu Hou, Ziniu Hu, Yujiang Li, Rui Lu, Jie Tang, and Yuxiao Dong. TreeRL: LLM reinforcement learning with on-policy tree search. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12355–12369, Vienna, Austria, July 2025.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: few-shot learning with retrieval augmented language models. *J. Mach. Learn. Res.*, 24, January 2023.

Pengcheng Jiang, Jiacheng Lin, Lang Cao, Runchu Tian, SeongKu Kang, Zifeng Wang, Jimeng Sun, and Jiawei Han. Deepretrieval: Hacking real search engines and retrievers with large language models via reinforcement learning. In *Second Conference on Language Modeling*, 2025.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan O Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training LLMs to reason and leverage search engines with reinforcement learning. In *Second Conference on Language Modeling*, 2025a.

Jiajie Jin, Yutao Zhu, Zhicheng Dou, Guanting Dong, Xinyu Yang, Chenghao Zhang, Tong Zhao, Zhao Yang, and Ji-Rong Wen. Flashrag: A modular toolkit for efficient retrieval-augmented generation research. In Guodong Long, Michale Blumestein, Yi Chang, Liane Lewin-Eytan, Zi Helen Huang, and Elad Yom-Tov (eds.), *Companion Proceedings of the ACM on Web Conference 2025, WWW 2025, Sydney, NSW, Australia, 28 April 2025 - 2 May 2025*, pp. 737–740. ACM, 2025b.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, Vancouver, Canada, 2017.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466, 2019.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*, 2021.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. Search-o1: Agentic search-enhanced large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pp. 5420–5438, Suzhou, China, 2025.

Zihan Liang, Yufei Ma, Ben Chen, Zhipeng Qian, Huangyu Dai, Lingtao Mao, Xuxin Zhang, Chenyi Lei, and Wenwu Ou. Ig-search: Step-level information gain rewards for search-augmented reasoning. *arXiv preprint arXiv:2604.15148*, 2026.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. *arXiv preprint arXiv:2305.20050*, 2023.

David Lopez-Paz, Léon Bottou, Bernhard Schölkopf, and Vladimir Vapnik. Unifying distillation and privileged information. *arXiv preprint arXiv:1511.03643*, 2016.

Yufei Ma, Zihan Liang, Ben Chen, Zhipeng Qian, Huangyu Dai, Lingtao Mao, Xuxin Zhang, Chenyi Lei, and Wenwu Ou. Sd-search: On-policy hindsight self-distillation for search-augmented reasoning. *arXiv preprint arXiv:2605.18299*, 2026.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9802–9822, Toronto, Canada, 2023.

OpenAI. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pp. 5687–5711, Singapore, 2023.

Qwen. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2025.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Idan Shenfeld, Mehul Damani, Jonas Hübotter, and Pulkit Agrawal. Self-distillation enables continual learning. *arXiv preprint arXiv:2601.19897*, 2026.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, pp. 1279–1297, 2025.

Yaorui Shi, Sihang Li, Chang Wu, Zhiyuan Liu, Junfeng Fang, Hengxing Cai, An Zhang, and Xiang Wang. Search and refine during think: Facilitating knowledge refinement for improved retrieval-augmented reasoning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. MuSiQue: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554, 2022.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 10014–10037, Toronto, Canada, 2023.

Guoqing Wang, Sunhao Dai, Guangze Ye, Zeyu Gan, Wei Yao, Yong Deng, Xiaofeng Wu, and Zhenzhe Ying. Information gain-based policy optimization: A simple and effective approach for multi-turn search agents. In *The Fourteenth International Conference on Learning Representations*, 2026.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. Text embeddings by weakly-supervised contrastive pre-training. *arXiv preprint arXiv:2212.03533*, 2024.

Ziliang Wang, Xuhui Zheng, Kang An, Cijun Ouyang, Jialu Cai, Yuhang Wang, and Yichao Wu. Stepsearch: Igniting llms search ability via step-wise proximal policy optimization. *arXiv preprint arXiv:2505.15107*, 2025.

Peilin Wu, Mian Zhang, Kun Wan, Wentian Zhao, Kaiyu He, Xinya Du, and Zhiyu Chen. HiPRAG: Hierarchical process rewards for efficient agentic retrieval augmented generation. In *The Fourteenth International Conference on Learning Representations*, 2026.

Tianle Xia, Ming Xu, Lingxiang Hu, Yiding Sun, Wenwei Li, Linfang Shang, Liqun Liu, Peng Shu, Huan Yu, and Jie Jiang. Search-p1: Path-centric reward shaping for stable and efficient agentic rag training. *arXiv preprint arXiv:2602.22576*, 2026.

Teng Xiao, Yige Yuan, Hamish Ivison, Huaisheng Zhu, Faeze Brahman, Nathan Lambert, Pradeep Dasigi, Noah A. Smith, and Hannaneh Hajishirzi. Meta-reinforcement learning with self-reflection for agentic search. *arXiv preprint arXiv:2603.11327*, 2026.

Jun Xu, Xinkai Du, Yu Ao, Peilong Zhao, Yang Li, Ling Zhong, Lin Yuan, Zhongpu Bo, Xiaorui Wang, Mengshu Sun, Zhengke Gui, Dalong Zhang, Zhaoyang Wang, Qiwei Wang, Yangyang Hou, Zhiying Yin, Haofen Wang, Huajun Chen, Lei Liang, and Jun Zhou. Thinker: Training llms in hierarchical thinking for deep search via multi-turn interaction. *arXiv preprint arXiv:2511.07943*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2369–2380, Brussels, Belgium, 2018.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. *arXiv preprint arXiv:2210.03629*, 2022.

Yaocheng Zhang, Haohuan Huang, Zijun Song, Yuanheng Zhu, Qichao Zhang, Zijie Zhao, and Dongbin Zhao. Criticsearch: Fine-grained credit assignment for search agents via a retrospective critic. *arXiv preprint arXiv:2511.12159*, 2025a.

Zhenru Zhang, Chujie Zheng, Yangzhen Wu, Beichen Zhang, Runji Lin, Bowen Yu, Dayiheng Liu, Jingren Zhou, and Junyang Lin. The lessons of developing process reward models in mathematical reasoning. *arXiv preprint arXiv:2501.07301*, 2025b.

Siyan Zhao, Zhihui Xie, Mengchen Liu, Jing Huang, Guan Pang, Feiyu Chen, and Aditya Grover. Self-distilled reasoner: On-policy self-distillation for large language models. *arXiv preprint arXiv:2601.18734*, 2026.

A Related Work

A.1 Reinforcement Learning for Search Augmented Reasoning

Retrieval augmented generation equips language models with external knowledge beyond their parametric memory (Lewis et al., 2021; Borgeaud et al., 2022; Izacard et al., 2023). Early approaches typically retrieve before generation or interleave retrieval with reasoning through prompting (Yao et al., 2022; Trivedi et al., 2023; Li et al., 2025). More recent work trains the model itself to decide when and how to search. Search-R1 (Jin et al., 2025a) establishes a simple formulation in which search calls are embedded directly into the reasoning trajectory and the policy is optimized with GRPO using final answer correctness. ReSearch (Chen et al., 2025) and AutoRefine (Shi et al., 2025) further improve this paradigm through richer trajectory formats and retrieval oriented training signals. Search-E1 retains the standard outcome based GRPO formulation and focuses instead on whether the experience generated by this simple training loop can itself support further improvement.

A.2 Beyond Outcome Based Reinforcement Learning

A growing body of work augments outcome based reinforcement learning with additional supervision or training structure. Thinker (Xu et al., 2025) distills decomposed search behavior from a substantially larger teacher, while StepSearch (Wang et al., 2025) uses externally generated intermediate supervision. Other methods derive denser learning signals through process reward models or retrospective critics (Zhang et al., 2025b; Lightman et al., 2023; Zhang et al., 2025a), policy based step estimators (Wang et al., 2026; Feng et al., 2025), or manually designed retrieval rewards (Xia et al., 2026; Wu et al., 2026; Hao et al., 2025). A separate direction modifies exploration itself through tree structured sampling or multi stage optimization (Ding & Ye, 2025; Hou et al., 2025; Jiang et al., 2025). These approaches differ substantially in mechanism, but share the goal of extracting supervision beyond a single trajectory level outcome. Closest to our setting are methods that derive their additional signal from the policy itself rather than from an external system, either by conditioning a teacher on a hindsight summary of an entire rollout group and aligning the student at search query positions (Ma et al., 2026), or by scoring each retrieval with an information gain estimate and modulating query token advantages (Liang et al., 2026). Search-E1 differs from both in two respects. Consolidation is a separate stage alternated with reinforcement learning rather than an auxiliary term inside a single update, and the privileged context is the search skeleton of one efficient successful trajectory rather than a group level summary or a scalar score. Search-E1 studies a complementary question, namely whether such additional machinery is necessary when successful and unsuccessful behaviors already coexist in the policy's own rollout distribution.

A.3 Self Distillation and Learning with Privileged Information

Self distillation allows a model to provide supervision for itself, either through predictions from an earlier policy or through asymmetric conditioning between teacher and student (Agarwal et al., 2024; Shenfeld et al., 2026). The latter is closely related to learning with privileged information (Lopez-Paz et al., 2016), where the teacher observes information that is unavailable to the student at inference time. OPSD (Zhao et al., 2026) applies this principle to language model reasoning by conditioning the teacher on privileged reasoning information and distilling its token distribution into the standard policy. Search-E1 builds on this mechanism but obtains the privileged information from the search skeleton of a sibling trajectory generated by the policy's own reinforcement learning process, so the teacher never observes retrieved evidence or the final answer. More importantly, self distillation is not used as a terminal post training stage. It is alternated with GRPO, so that reinforcement learning generates new experience and self distillation consolidates useful behaviors across successive rounds. This repeated interaction between exploration and consolidation is the focus of Search-E1.

B Dataset Statistics

Table 4 summarizes the seven question answering benchmarks used in our experiments. All datasets are obtained from the FlashRAG collection (Jin et al., 2025b). Following prior work (Jin et al., 2025a; Shi et al., 2025), we form the training corpus by combining the training splits of NQ and HotpotQA, which yields $169{,}615$ question answer pairs. For evaluation we use the test split where one is available, namely NQ, TriviaQA, PopQA and Bamboogle, and the development split otherwise, namely HotpotQA, 2Wiki and MuSiQue.

Table 4: Statistics of the seven benchmarks used in this paper.

Dataset	*Split sizes*			**Eval split**
	Train	Dev	Test	
NQ	79,168	8,757	3,610	test
TriviaQA	78,785	8,837	11,313	test
PopQA	–	–	14,267	test
HotpotQA	90,447	7,405	–	dev
2Wiki	15,000	12,576	–	dev
MuSiQue	19,938	2,417	–	dev
Bamboogle	–	–	125	test

Question pool across rounds. The self evolution loop maintains a question pool $\mathcal{D}_r$ that is initialized to the full training corpus. A single GRPO round consumes far fewer instances than the corpus contains, so each round visits only a subset of the pool. Rollout mining and the recycling rules of Equations 10 and 11 are applied to the questions visited in that round, and questions that were never visited remain in the pool unchanged. Round wise pool statistics are reported in Appendix H.

Validation and calibration sets. Two auxiliary sets are used during training and never contribute to any reported result. The validation set contains $1{,}000$ questions held out from the NQ and HotpotQA training splits, 500 from each, and is excluded from the question pool and from every evaluation split. All checkpoint selection, including the entropy aware handoff rule of Section 3.4, is performed on this set, so no benchmark test or development split is consulted during training. The calibration set contains 512 additional held out questions and is kept fixed across all rounds. It is used only to measure mean token entropy and rollout diversity, and no gradient is computed on it.

C Complete Training Configuration

Table 5 lists the complete configuration used in our experiments. Values not listed follow the default configuration of veRL (Sheng et al., 2025). The GRPO block is inherited unchanged from the outcome reward baselines, so the only quantities introduced by Search-E1 are the self evolution schedule and the OPSD settings. No per dataset tuning is performed, and the same configuration is used for the Base and Instruct variants at both scales.

Table 5: Complete configuration for Search-E1 training.

Hyperparameter	**Value**
Optimization	
Framework	veRL
Hardware	$8 \times$ H200
Training batch size	256
Learning rate, GRPO and OPSD	1×10^{-6}
KL coefficient β	0.001
Clip ratio ϵ	0.2
Max response length	4096 tokens
Rollout and reward	
Group size G	5
Rollout temperature	1.0
Max search calls per rollout	4
Outcome reward	Exact Match
Retrieval	
Retriever	E5-base-v2
Corpus	Wikipedia (Dec. 2018)
Passages per query	3
Self evolution schedule	
Cycles, 3B / 7B	3 / 2
GRPO steps, first round	200
GRPO steps, later rounds	100
Mining rollouts per visited question K	5
Reference selection	fewest search calls, tie by skeleton length
Entropy retention threshold	0.85
OPSD	
Divergence	forward KL
Pointwise clip τ_{clip}	10
Alignment scope	$\mathcal{A}_\tau$
Steps per round	100
Checkpoint interval	20 steps
Privileged context max length	512 tokens
Teacher	frozen copy of π_τ^{RL}

Schedule. The first GRPO round uses 200 steps, matching the budget of the outcome reward baselines we compare against, and each later round uses 100 steps. Every OPSD round runs for 100 steps and stores a checkpoint every 20 steps, which yields six candidates including the initialization. The checkpoint selected by the entropy aware handoff rule initializes the next GRPO round, and the remaining OPSD steps are discarded. Appendix E reports the full accounting of executed steps and wall clock time, including the discarded tail.

D PRIVILEGED CONTEXT CONSTRUCTION

Section 2.2 restricts the privileged context to the search skeleton of the reference trajectory. This appendix specifies the resulting teacher input, isolates the contribution of each part of the reference that we discard, and reports the behavioral evidence behind the design.

Teacher input layout. Figure 4 renders one teacher input as the model receives it. The teacher sees three blocks in causal order, namely the task instruction with the question, the privileged skeleton, and the prefix of the student rollout up to the supervised position. The student sees the same input with the skeleton block removed. The difference between the two contexts at every supervised position is therefore exactly one block, and every supervised position has causal access to the whole skeleton. At inference time the skeleton block is absent and the two views collapse into a single policy.

(1) Instruction and question, identical for student and teacher

```
You are a helpful assistant.  Answer the question by reasoning
and searching when needed.  Question:  What is the date of death
of Alexander of Masovia's father?
```

(2) Privileged skeleton, teacher only

```
[Reference search plan]
<search> Alexander of Masovia father </search> ->
<search> Siemowit IV Duke of Masovia date of death </search>
```

(3) Student rollout prefix, identical for student and teacher

```
<think> First identify the father, then look up his death
date.  </think> <search> Alexander of Masovia father </search>
<information> ...  </information> <think> Now I need the
death date of Siemowit IV. </think> <search>
```

(4) Supervised positions, forward KL applied over $\mathcal{A}_{\tau^{\mathrm{stu}}}$

Figure 4: Teacher input layout for one supervised position. Block (2) contains only the `<search>` spans of the reference trajectory, in their original order. Intermediate reasoning, retrieved passages and the final answer of the reference are removed by the masking operator $\mathcal{M}$. The student input is identical with block (2) deleted, so the difference between the two contexts is the shaded block.

What the teacher should and should not see. We vary the content of the privileged block while keeping the rest of the pipeline fixed, and evaluate each variant under the full three cycle schedule with Qwen2.5-3B-Instruct. Beyond Exact Match we report two behavioral quantities measured on the evaluation benchmarks, namely the average number of search calls per rollout and the search quality, defined as the fraction of retrieval calls whose returned passages contain the gold answer. We also report the average length of the privileged block in tokens. Table 6 reports the results.

Table 6: Privileged context construction on Qwen2.5-3B-Instruct. Single and Multi denote the average Exact Match over the three single hop and four multi hop benchmarks. Calls is the average number of search calls per rollout and Quality is the fraction of retrieval calls returning the gold answer. Length is the average size of the privileged block in tokens.

Configuration	Single	Multi	Avg.	Calls	Quality	Length
Search skeleton (Search-E1)	0.520	0.380	**0.440**	2.12	54.3	41
Skeleton with shuffled query order	0.515	0.368	0.431	2.09	51.8	41
Skeleton with final answer appended	0.509	0.362	0.425	1.91	48.7	47
Skeleton with retrieved passages	0.512	0.365	0.428	1.74	50.1	1183
Full trajectory	0.506	0.356	0.420	1.38	46.2	1930
No privileged context	0.507	0.365	0.426	2.05	52.0	0

Three observations follow. First, exposing the answer is harmful rather than merely redundant. Appending the final answer of the reference costs 1.5 points, and passing the full trajectory costs

2.0 points, which places it below the 0.426 obtained without any distillation at all. The behavioral columns explain why. As more of the reference becomes visible, the teacher can predict the gold answer from its own context without issuing a query, so its distribution at search positions concentrates on continuations that terminate retrieval early. Search calls per rollout fall from 2.12 to 1.38 and search quality falls by 8.1 points. The student inherits this behavior and answers on weaker evidence. A teacher that already knows the answer is therefore not a stronger teacher, and the value of the privileged context lies in the search plan rather than in the outcome.

Second, the ordering of the skeleton carries information. Shuffling the queries of the reference while keeping the same set costs 0.9 points, with the loss concentrated on the multi hop benchmarks, where the second query is typically conditioned on the entity resolved by the first. The skeleton is thus read by the teacher as a plan rather than as a bag of keywords.

Third, the skeleton is also the cheapest option. It averages 41 tokens, against $1{,}930$ tokens for the full trajectory, because retrieved passages dominate the length of a search augmented trajectory. Restricting the privileged context to the skeleton therefore removes the leakage, improves accuracy, and keeps the teacher forward pass close to the cost of the student pass.

The last row is a degenerate control in which the teacher and the student receive identical contexts. The forward KL term is then zero by construction and training reduces to GRPO with the same recycling schedule, which recovers the GRPO only entry of Table 3 and confirms that the gain is attributable to the asymmetry of the conditioning rather than to the additional optimization stage.

E Training Budget and Compute Accounting

Search-E1 executes more optimizer steps than the single stage baselines it is compared against, so a natural concern is whether the reported gains reflect the alternation of exploration and consolidation or simply a larger training budget. This appendix reports the full cost of a self evolution run and compares Search-E1 against a control matched on executed optimizer steps.

Cost of one self evolution run. On 8 $\times$ H200 with Qwen2.5-3B and a batch size of 256, one GRPO step takes 60 seconds and one OPSD step takes 66 seconds. Rollout generation accounts for 44 seconds of both, so the components specific to OPSD, namely the teacher forward pass over the privileged context and the token level divergence, add roughly one tenth of a step. Table 7 reports the resulting breakdown. The accounting includes the rollout mining stages and the checkpoint evaluations required by the handoff rule, and it charges each OPSD round for all 100 executed steps even though the tail after the selected checkpoint is discarded.

Table 7: Wall clock breakdown of one complete Search-E1 run on Qwen2.5-3B, 8 $\times$ H200, batch size 256. Steps counts optimizer steps. Mining and checkpoint selection perform generation and evaluation only.

Stage	Steps	Wall clock (h)
GRPO round 1	200	3.33
Rollout mining 1	–	2.44
OPSD round 1	100	1.83
GRPO round 2	100	1.67
Rollout mining 2	–	1.22
OPSD round 2	100	1.83
GRPO round 3	100	1.67
Rollout mining 3	–	1.22
OPSD round 3	100	1.83
Checkpoint selection	–	0.30
Total	700	17.36

The complete run takes 17.4 hours, against 3.5 hours for a 200 step outcome reward baseline trained under the same hardware, batch size, rollout configuration and retriever. Two properties of this cost are worth noting. It is entirely contained within the standard reinforcement learning loop, since no external model is queried and no annotation phase is performed, and it leaves inference unchanged, because the privileged context and the teacher forward pass are removed at test time. Mining accounts for 28 percent of the total, and its cost is proportional to the number of questions visited in the preceding round rather than to the size of the corpus.

Step matched comparison. To separate the effect of alternation from the effect of budget, we train a control that runs GRPO alone for 700 optimizer steps, which is the total number of steps executed by Search-E1, under the identical recycling schedule and starting from the same round one checkpoint. This control costs 11.8 hours, so Search-E1 spends 1.47 times its wall clock. Figure 5 plots average Exact Match against cumulative optimizer steps for both runs.

The two runs coincide at step 200 by construction and then diverge. The control improves to 0.426 at step 400, peaks at 0.427 at step 500, and declines to 0.422 by step 700, whereas Search-E1 reaches 0.440. The decline of the control is not caused by a loss of exploration capacity, since its mean token entropy on the calibration set still retains 0.93 of its step 200 value at step 700, well above the level at which the handoff rule intervenes. Additional outcome reward optimization on an already recycled pool therefore stops converting rollouts into improvement, while the same rollouts still carry contrasts that consolidation can extract.

Comparison with externally supervised pipelines. The methods that Search-E1 outperforms in Tables 1 and 2 carry costs that their reported training time does not include. Thinker (Xu et al., 2025) runs inference with a 72B teacher over its training set and adds a supervised fine tuning stage before reinforcement learning, and StepSearch (Wang et al., 2025) depends on sub question annotations generated by a commercial model over the full training corpus, so its cost scales with dataset size in

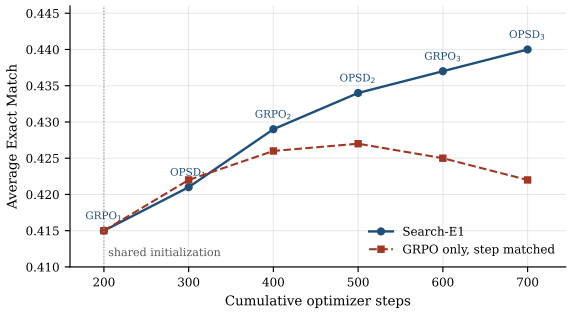

Figure 5: Average Exact Match against cumulative optimizer steps on Qwen2.5-3B-Instruct. Both runs share the first 200 GRPO steps and the same recycling schedule. The Search-E1 markers report the checkpoint selected at the end of each stage. The control continues with GRPO alone for the same total number of steps.

both wall clock and monetary terms. Search-E1 replaces these external stages with additional passes of the same training loop, which we regard as the more reproducible point on the cost and quality frontier for practitioners without access to a much larger teacher or an annotation budget.

F Full Results Across Self Evolution Rounds

Figure 2 in the main text reports only the seven benchmark average after each stage. This appendix gives the complete per benchmark results for every stage of every backbone. GRPO_r denotes the policy at the end of the r th exploration round. OPSD_r denotes the checkpoint selected by the entropy aware handoff rule at the end of the r th consolidation round, which is also the checkpoint that initializes the following GRPO round. The last row of each table reproduces the corresponding entry of Table 1 or Table 2.

Table 8: Exact Match after every stage of Search-E1 training with Qwen2.5-3B-Base. GRPO_r denotes the policy at the end of the r th exploration round and OPSD_r the checkpoint selected by the entropy aware handoff rule at the end of the r th consolidation round. The final row reproduces the corresponding entry of the main results table.

	Single-Hop			*Multi-Hop*				
Stage	NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	**Avg.**
GRPO_1	0.461	0.623	0.460	0.405	0.400	0.165	0.377	0.413
OPSD_1	0.466	0.628	0.465	0.411	0.406	0.171	0.386	0.419
GRPO_2	0.471	0.633	0.470	0.420	0.417	0.179	0.399	0.427
OPSD_2	0.475	0.637	0.474	0.425	0.422	0.184	0.407	0.432
GRPO_3	0.477	0.639	0.476	0.429	0.425	0.187	0.412	0.435
OPSD_3	**0.479**	**0.641**	**0.478**	**0.432**	**0.429**	**0.190**	**0.416**	**0.438**

Table 9: Exact Match after every stage of Search-E1 training with Qwen2.5-3B-Instruct. GRPO_r denotes the policy at the end of the r th exploration round and OPSD_r the checkpoint selected by the entropy aware handoff rule at the end of the r th consolidation round. The final row reproduces the corresponding entry of the main results table.

	Single-Hop			*Multi-Hop*				
Stage	NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	**Avg.**
GRPO_1	0.456	0.609	0.444	0.399	0.406	0.168	0.423	0.415
OPSD_1	0.461	0.613	0.448	0.406	0.412	0.174	0.433	0.421
GRPO_2	0.466	0.618	0.453	0.415	0.423	0.182	0.446	0.429
OPSD_2	0.470	0.622	0.457	0.420	0.429	0.187	0.453	0.434
GRPO_3	0.472	0.624	0.459	0.424	0.432	0.190	0.458	0.437
OPSD_3	**0.474**	**0.626**	**0.461**	**0.427**	**0.436**	**0.193**	**0.464**	**0.440**

Table 10: Exact Match after every stage of Search-E1 training with Qwen2.5-7B-Base. GRPO_r denotes the policy at the end of the r th exploration round and OPSD_r the checkpoint selected by the entropy aware handoff rule at the end of the r th consolidation round. The final row reproduces the corresponding entry of the main results table.

	Single-Hop			*Multi-Hop*				
Stage	NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	**Avg.**
GRPO_1	0.483	0.663	0.485	0.440	0.412	0.196	0.478	0.451
OPSD_1	0.488	0.668	0.490	0.449	0.422	0.204	0.492	0.459
GRPO_2	0.497	0.677	0.499	0.462	0.436	0.216	0.510	0.471
OPSD_2	**0.503**	**0.683**	**0.505**	**0.472**	**0.447**	**0.225**	**0.525**	**0.480**

Three patterns hold across all four backbones. Every consolidation stage improves on the exploration stage that precedes it, and every exploration stage improves on the consolidation stage that precedes it, so neither stage merely undoes the other. The per stage gains are larger on the multi hop benchmarks throughout, which is consistent with the interpretation that a privileged search plan is most informative when the question requires several dependent queries. The gains contract from

Table 11: Exact Match after every stage of Search-E1 training with Qwen2.5-7B-Instruct. GRPO_r denotes the policy at the end of the r th exploration round and OPSD_r the checkpoint selected by the entropy aware handoff rule at the end of the r th consolidation round. The final row reproduces the corresponding entry of the main results table.

	Single-Hop			*Multi-Hop*				
Stage	NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	**Avg.**
GRPO_1	0.464	0.671	0.474	0.433	0.416	0.203	0.482	0.449
OPSD_1	0.470	0.677	0.480	0.443	0.427	0.212	0.497	0.458
GRPO_2	0.480	0.687	0.490	0.458	0.444	0.226	0.519	0.472
OPSD_2	**0.487**	**0.694**	**0.497**	**0.469**	**0.455**	**0.236**	**0.534**	**0.482**

round to round, and the third cycle at the 3B scale contributes less than half of what the first contributes. We did not observe a round at which performance decreased, but we also did not extend the schedule beyond the point at which the marginal gain fell below the seed level variation reported in Appendix O.

G ALTERNATIVES TO CONSOLIDATION

Search-E1 consumes its own successful rollouts through on policy self distillation. Two simpler alternatives exist. The experience can be consumed by supervised fine tuning on the mined trajectories, and the consolidation stage can be applied once instead of being alternated with exploration. Table 12 compares these under the full three cycle schedule with Qwen2.5-3B-Instruct.

For the supervised variants, each OPSD round is replaced by 100 steps of supervised fine tuning at the same learning rate on the same mined references, with the loss taken over policy generated positions only, and the resulting checkpoints are passed to the next round by the same entropy aware rule. The two supervised variants differ in the target sequence, namely the complete successful trajectory or its search skeleton alone. For the single stage variants, the entire consolidation budget of 300 steps is spent at one point in the schedule, either after the first GRPO round or after all three. The last column reports the entropy retention measured on the calibration set at the end of the consolidation stage, before checkpoint selection.

Table 12: Alternatives to on policy self distillation on Qwen2.5-3B-Instruct. Single and Multi denote the average Exact Match over the three single hop and four multi hop benchmarks. Entropy is the entropy retention on the calibration set at the end of the consolidation stage, before checkpoint selection.

Configuration	**Single**	**Multi**	**Avg.**	**Entropy**
OPSD, alternating (Search-E1)	0.520	0.380	**0.440**	0.69
Distillation once after the first round	0.513	0.368	0.430	0.69
Distillation once after all GRPO rounds	0.514	0.370	0.432	0.68
SFT on mined trajectories, alternating	0.512	0.365	0.428	0.52
SFT on mined skeletons, alternating	0.509	0.361	0.424	0.58
No consolidation (GRPO only)	0.507	0.365	0.426	–

Supervised fine tuning recovers only a small part of the benefit. Fitting the complete mined trajectory reaches 0.428 and fitting the skeleton alone reaches 0.424, against 0.440 for distillation. Two differences account for the gap. The supervised objective is evaluated on the reference sequence, so it supplies no signal at the positions the student actually visits when it departs from that sequence, whereas the distillation objective is evaluated along the student's own rollout. The supervised objective also drives the policy toward a single realization rather than toward a distribution, and the entropy column shows the consequence. Supervised fine tuning removes roughly half of the policy entropy within one hundred steps, against thirty one percent for distillation, so under the same handoff constraint it is forced to stop far earlier and consolidates correspondingly less. This is the same mechanism examined in Appendix K, observed here as a property of the objective rather than of the number of steps.

Applying distillation once rather than alternating is also worse. Spending the entire consolidation budget after the first GRPO round reaches 0.430, and spending it after all three rounds reaches 0.432. Neither approaches the alternating schedule. When consolidation happens only at the beginning, the behaviors it absorbs are those of an early policy and become stale as exploration continues. When it happens only at the end, the exploration rounds never benefit from the behaviors that earlier rounds discovered, so each round rediscovers ground that a consolidated policy would already hold. The value of consolidation therefore lies in its position within the loop and not only in the objective it optimizes.

H Question Recycling and Pool Composition

Section 2.3 retains for the next GRPO round every visited question that is not uniformly solved, so all failure questions and mixed outcome questions are carried forward while uniformly solved ones are dropped. This appendix reports how the rollout pool is composed across rounds and compares this rule against three alternatives.

Pool composition across rounds. After each GRPO round we sample $K = 5$ trajectories for every question visited during that round and partition the questions by the number of correct trajectories $c_r(q)$. Table 13 reports the resulting composition on Qwen2.5-3B-Instruct.

Table 13: Rollout pool composition after each GRPO round on Qwen2.5-3B-Instruct. Percentages are taken over the questions visited in that round. References counts the questions that supply a privileged reference for the following OPSD round. Pool size is the number of questions remaining after uniformly solved ones are removed.

Round	Visited	$c_r{=}0$ (%)	$0{<}c_r{<}K$ (%)	$c_r{=}K$ (%)	References	Pool size
1	51,200	38.0	44.0	18.0	31,744	160,399
2	25,600	31.0	46.0	23.0	17,664	154,511
3	25,600	27.0	47.0	26.0	18,688	147,855

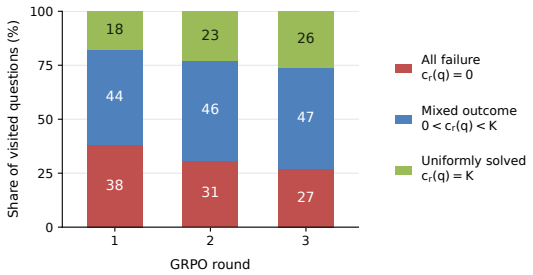

Figure 6: Composition of the rollout pool after each GRPO round on Qwen2.5-3B-Instruct. The share of uniformly solved questions grows across rounds while the share of all failure questions shrinks, so the two groups that the recycling rule retains change in opposite directions as training proceeds.

Two trends are visible, and Figure 6 shows the same composition. The fraction of uniformly solved questions grows from 18.0 to 26.0 percent as the policy improves, so an increasing share of every round would otherwise be spent on questions from which GRPO can extract no advantage signal, since a group in which every trajectory succeeds yields a zero normalized advantage exactly as one in which every trajectory fails. At the same time the fraction of all failure questions falls from 38.0 to 27.0 percent, which means the questions that supply no privileged reference are also the ones that shrink fastest. The mixed outcome group, which contributes to both stages, remains close to half of every round throughout training.

Which questions should be retained. We compare four recycling rules under the full three cycle schedule, keeping everything else fixed. The alternatives retain only mixed outcome questions, retain only all failure questions, or retain every visited question regardless of outcome. Table 14 reports the results.

Retaining every visited question costs 0.8 points. Uniformly solved questions produce a zero advantage under group normalization, so they consume rollout budget without contributing gradient, and their share grows with every round. Removing them concentrates later exploration on the questions where search behavior can still change.

Discarding either of the two informative groups is worse than keeping both. Retaining mixed outcome questions alone costs 0.9 points, and retaining all failure questions alone costs 1.4 points. The

Table 14: Question recycling strategies on Qwen2.5-3B-Instruct. Single and Multi denote the average Exact Match over the three single hop and four multi hop benchmarks.

Retained for the next round	**Single**	**Multi**	**Avg.**
Failure and mixed outcome (Search-E1)	0.520	0.380	**0.440**
Mixed outcome only	0.512	0.371	0.431
All failure only	0.505	0.367	0.426
Retain every visited question	0.510	0.374	0.432

two groups play different roles. Mixed outcome questions are the only source of privileged references among the questions that also remain useful for exploration, so dropping them starves the consolidation stage. All failure questions supply no reference in the current round, yet they are the questions on which the policy has the most room to improve, and Table 13 shows that a substantial part of them converts into mixed outcome questions in later rounds. Keeping both groups is therefore not a compromise between two criteria but a consequence of the two stages consuming different parts of the same pool.

I PRIVILEGED REFERENCE SELECTION

By default Search-E1 selects, among the successful rollouts of a question, the one with the fewest search calls, breaking ties by the total length of its skeleton. Table 15 compares this rule against five alternatives under the full three cycle schedule with Qwen2.5-3B-Instruct. The last column reports the average number of search calls in the selected reference, which is the quantity the selection rule most directly controls.

Table 15: Privileged reference selection on Qwen2.5-3B-Instruct. Reference calls is the average number of search calls in the selected reference trajectory.

Selection rule	Single	Multi	Avg.	Reference calls
Fewest search calls (Search-E1)	0.520	0.380	**0.440**	1.68
Shortest skeleton in tokens	0.518	0.377	0.437	1.79
Random successful trajectory	0.516	0.374	0.435	2.24
Longest successful trajectory	0.510	0.372	0.431	3.12
Most search calls	0.508	0.370	0.429	3.31
Random rollout regardless of outcome	0.503	0.358	0.420	2.41

Performance decreases monotonically as the selected reference becomes less efficient, from 0.440 at 1.68 average calls to 0.429 at 3.31. A successful trajectory that reaches the answer after several redundant queries still demonstrates a path to the answer, but the redundant steps are part of the plan the teacher conditions on, so the teacher rewards them at the student's search positions as readily as the productive ones. Selecting by skeleton length rather than by call count performs almost as well at 0.437, which indicates that the useful notion of efficiency is the number of retrieval decisions rather than verbosity, although the two are correlated.

The final row removes the correctness requirement entirely and selects a uniformly random rollout as the reference. This reaches 0.420, below the 0.426 obtained without any consolidation, and is the only variant in this table that is actively harmful. Conditioning on an unsuccessful plan makes the teacher a worse predictor than the student at exactly the positions the objective targets, so the distillation term transfers failure modes rather than removing them. Correctness is therefore not a convenience for filtering but the property that makes the asymmetry between teacher and student informative, which also explains why the method inherits the requirement for automatically checkable answers.

J Distillation Objective and Alignment Scope

The consolidation stage makes two choices, namely the divergence between the teacher and student distributions and the set of positions over which it is summed. We vary each while keeping everything else fixed, under the full three cycle schedule with Qwen2.5-3B-Instruct. Table 16 reports both the final Exact Match and the entropy retention at the end of the consolidation round, because the two are not ordered in the same way and the second determines how much of the round the handoff rule can use.

Table 16: Divergence and alignment scope on Qwen2.5-3B-Instruct. Entropy is the entropy retention on the calibration set at the end of the second consolidation round, before checkpoint selection.

Configuration	Single	Multi	Avg.	Entropy
Divergence, scope fixed to $\mathcal{A}_\tau$				
Forward KL with pointwise clipping (Search-E1)	0.520	0.380	**0.440**	0.69
Forward KL without clipping	0.514	0.372	0.433	0.66
Jensen-Shannon divergence	0.517	0.375	0.436	0.74
Reverse KL	0.512	0.365	0.428	0.41
Mean squared error on logits	0.506	0.358	0.421	0.58
Alignment scope, divergence fixed to clipped forward KL				
All policy generated positions (Search-E1)	0.520	0.380	**0.440**	0.69
Reasoning and search positions	0.518	0.377	0.437	0.71
Search query positions only	0.513	0.375	0.434	0.79
Answer positions only	0.509	0.362	0.425	0.83

Divergence. Reverse KL is the weakest of the three divergence choices at 0.428, and it is also the one that sharpens the policy most, retaining only 0.41 of the entropy within one hundred steps. Its mode seeking behavior collapses the student onto the teacher's most likely continuation, which suppresses precisely the variation among sibling rollouts that the next exploration round needs. This connects the objective to the evolvability analysis of Section 3.4, since a divergence that sharpens faster leaves the handoff rule a smaller usable portion of each round. The Jensen-Shannon divergence is the closest competitor at 0.436 and preserves the most entropy, but it is bounded and symmetric, so at the many positions where teacher and student already agree it attenuates an already small signal, and the resulting update is slower without being better targeted. Removing the pointwise clip from the forward KL costs 0.7 points and produces visibly noisier training curves, because a small number of vocabulary entries with very large teacher probability ratios dominate individual updates. Mean squared error on the raw logits performs worst, as it treats every coordinate as equally important and ignores the simplex structure that the divergences respect.

Alignment scope. Narrowing the scope to the search query positions costs 0.6 points. This is a smaller effect than any of the divergence changes, which is expected since the query positions carry most of the behavioral difference the privileged plan induces, but it is consistent. The privileged context is a plan rather than an answer, so it also changes how the policy reasons between retrievals, and aligning the reasoning positions transfers part of that. Restricting the objective to the answer positions alone is the weakest configuration at 0.425, close to no consolidation, confirming that the transferred behavior concerns how evidence is gathered rather than how the final string is produced.

K ENTROPY AND ROLLOUT DIVERSITY

Section 3.4 uses mean token entropy as the criterion for passing a checkpoint to the next exploration round. Entropy is a per position quantity, so it is not obvious that it tracks the trajectory level diversity that GRPO actually consumes. This appendix measures both on the same checkpoints of the second consolidation round of Qwen2.5-3B-Instruct.

For each checkpoint we sample $K = 5$ rollouts for every question of the calibration set and report four quantities. Distinct trajectories counts how many of the five rollouts are pairwise different after normalizing whitespace. Distinct first queries counts how many different opening queries appear among the five. Pairwise distance is the mean normalized edit distance between the search skeletons of two sibling rollouts. Reference coverage is the fraction of questions for which at least one of the five rollouts is correct, which is the fraction that would supply a privileged reference in the next mining stage.

Table 17: Entropy and rollout diversity over the second consolidation round of Qwen2.5-3B-Instruct. All quantities are measured on the fixed calibration set with $K = 5$ rollouts per question. Entropy is expressed relative to the value at the beginning of the round.

OPSD step	Entropy	Distinct traj.	Distinct first queries	Pairwise dist.	Coverage
0	1.00	4.31	3.72	0.62	0.69
20	0.95	4.18	3.61	0.60	0.70
40	0.91	4.02	3.45	0.57	0.71
60	0.86	3.84	3.28	0.54	0.71
80	0.78	3.46	2.90	0.47	0.70
100	0.69	3.02	2.44	0.39	0.68

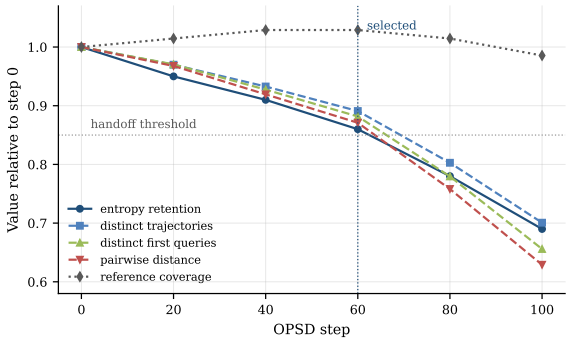

Figure 7: Entropy retention and rollout diversity over the second consolidation round of Qwen2.5-3B-Instruct. Diversity quantities are normalized by their value at the beginning of the round. The three diversity measures track entropy closely, while reference coverage stays nearly flat until entropy falls below the handoff threshold.

The three diversity measures track entropy closely and decline faster than linearly in the second half of the round. Between step 60 and step 100 the entropy retention falls from 0.86 to 0.69, while the number of distinct trajectories among five falls from 3.84 to 3.02 and the number of distinct opening queries falls from 3.28 to 2.44. At that point roughly half of the sampled rollouts for a question begin with the same query, so a group of five carries substantially less information than its size suggests. Since the group normalized advantage of Equation 1 is defined by the spread of outcomes within a group, this directly reduces the signal available to the next exploration round.

Reference coverage behaves differently. It rises slightly through the first half of the round as consolidation makes the policy more reliable, peaks at 0.71, and then falls back to 0.68 once the collapse in diversity outweighs the improvement in per rollout accuracy. The checkpoint selected by the handoff rule sits at the peak of this quantity, which is a useful coincidence rather than a criterion we optimize, since coverage can only be measured after the following mining stage while entropy is available during the round itself.

L ENTROPY AWARE HANDOFF THRESHOLD

The handoff rule keeps checkpoints whose entropy retention is at least 0.85 and selects the best of these by validation Exact Match. Table 18 varies this threshold. For each value we report the checkpoint the rule selects in the second consolidation round, its immediate validation Exact Match, the Exact Match reached by the exploration round initialized from it, and the average Exact Match at the end of the complete schedule.

Table 18: Entropy retention threshold on Qwen2.5-3B-Instruct. Selected is the OPSD step chosen by the rule in the second consolidation round. Immediate and Next GRPO are validation Exact Match before and after the following exploration round. Final is the seven benchmark average at the end of the complete schedule.

Threshold	Selected	Entropy	Immediate	Next GRPO	Final
0.65	100	0.69	0.438	0.432	0.434
0.75	80	0.78	0.436	0.435	0.437
0.85	60	0.86	0.434	0.437	**0.440**
0.90	40	0.91	0.433	0.436	0.438
0.95	20	0.95	0.431	0.434	0.435

The final column is single peaked. A threshold of 0.65 admits every checkpoint, so the rule reduces to selecting the best immediate validation accuracy and recovers the variant reported as w/o entropy aware handoff in Table 3. A threshold of 0.95 is the opposite failure and stops consolidation after twenty steps, discarding most of the benefit that the stage would have provided. Between these, the final result varies by 0.6 points across the range from 0.75 to 0.90, so the rule does not require precise tuning, but the two extremes cost 0.5 and 0.6 points respectively. A threshold of 0.80 selects the same checkpoint as 0.85 in this round and is omitted from the table.

The immediate and final columns are ordered inversely over the top three rows, which is the central observation of Section 3.4 restated as a sweep. Selecting on immediate accuracy alone is not merely suboptimal, it selects in the wrong direction, because within one consolidation round accuracy and diversity move against each other and only the second is consumed by the round that follows.

M SENSITIVITY TO ROLLOUT SIZES

Two rollout sizes govern the loop, namely the GRPO group size G and the number of rollouts K sampled per visited question during mining. Table 19 varies each while holding the other at its default, under the full three cycle schedule with Qwen2.5-3B-Instruct. Cost is the total wall clock of the run relative to the default configuration. Coverage is the fraction of visited questions that supply a privileged reference after the first round.

Table 19: Sensitivity to the GRPO group size G and the mining size K on Qwen2.5-3B-Instruct. Cost is total wall clock relative to the default configuration. Coverage is the fraction of visited questions with at least one successful rollout after the first round.

Configuration	Single	Multi	Avg.	Coverage	Cost
$G = 3$	0.515	0.368	0.431	–	0.72
$G = 5$ (Search-E1)	0.520	0.380	**0.440**	–	1.00
$G = 8$	0.521	0.381	0.441	–	1.46
$K = 3$	0.513	0.375	0.434	0.55	0.83
$K = 5$ (Search-E1)	0.520	0.380	**0.440**	0.62	1.00
$K = 10$	0.522	0.382	0.442	0.71	1.42

Both parameters show clear returns up to the default and very little beyond it. Reducing the group size to three costs 0.9 points, since a group of three frequently contains either no successful trajectory or no unsuccessful one, in which case the normalized advantage carries no information and no reference is mined. Increasing it to eight adds 0.1 points for forty six percent more compute. Reducing the mining size to three costs 0.6 points and lowers reference coverage from 0.62 to 0.55, so a larger share of questions receives no consolidation signal in a given round. Increasing it to ten raises coverage to 0.71 but adds only 0.2 points, because the additional references are drawn from questions the policy solves rarely, whose successful rollouts tend to be long and inefficient and are therefore poor candidates under the selection rule of Appendix I. We keep $G = K = 5$ throughout, which also matches the group size used by the outcome reward baselines and keeps the exploration stage identical to theirs.

N SEARCH BEHAVIOR ACROSS ROUNDS

A consolidation stage that improves accuracy could in principle do so by changing how much the policy retrieves rather than how well it retrieves. Two degenerate outcomes are possible, namely learning to issue more queries in the hope of covering the answer, and learning to skip retrieval and answer from parametric knowledge. Table 20 reports three behavioral quantities after every stage of the 3B Instruct run, measured on the evaluation benchmarks.

Table 20: Search behavior after every stage of Search-E1 training with Qwen2.5-3B-Instruct, measured on the seven evaluation benchmarks. Calls is the average number of search calls per rollout, Quality is the fraction of retrieval calls whose passages contain the gold answer, and Length is the average number of generated tokens per rollout.

Stage	Calls	Quality (%)	Length	Avg. EM
$GRPO_1$	2.03	49.1	412	0.415
$OPSD_1$	2.06	50.4	398	0.421
$GRPO_2$	2.09	51.9	405	0.429
$OPSD_2$	2.11	52.8	392	0.434
$GRPO_3$	2.11	53.6	397	0.437
$OPSD_3$	2.12	54.3	389	0.440

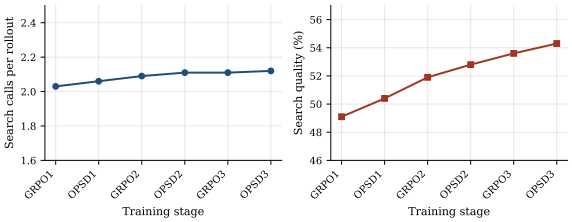

Figure 8: Search behavior across the stages of Search-E1 training with Qwen2.5-3B-Instruct. The number of search calls per rollout stays nearly constant while the fraction of retrievals returning the gold answer rises monotonically, so the improvement comes from better queries rather than from more of them.

Neither degenerate outcome occurs. The number of search calls per rollout moves from 2.03 to 2.12 over the entire schedule, a change of less than five percent, while search quality rises from 49.1 to 54.3 percent and the average Exact Match rises by 2.5 points. The improvement therefore comes from the content of the queries rather than from their number. Generated length falls slightly, from 412 to 389 tokens, which is consistent with the selection rule favoring efficient references, but the change is small enough that it cannot account for the accuracy gain. The consolidation stages contribute most of the movement in search quality, with an average rise of 1.0 points per OPSD stage against 1.2 points per GRPO stage, so both stages improve retrieval and neither is merely stabilizing the other.

O Robustness Across Random Seeds

We repeat the complete three cycle schedule with three random seeds on Qwen2.5-3B-Instruct, together with the GRPO only control under identical settings. Table 21 reports the mean and standard deviation across seeds.

Table 21: Exact Match mean and standard deviation across three random seeds on Qwen2.5-3B-Instruct. The GRPO only control uses the same schedule, question recycling and evaluation protocol.

	Single-Hop			*Multi-Hop*				
Method	NQ	TriviaQA	PopQA	HotpotQA	2Wiki	MuSiQue	Bamboogle	**Avg.**
Search-E1	0.474 $\pm$0.007	0.626 $\pm$0.005	0.461 $\pm$0.008	0.427 $\pm$0.010	0.436 $\pm$0.012	0.193 $\pm$0.009	0.464 $\pm$0.024	0.440 $\pm$0.006
GRPO only	0.462 $\pm$0.008	0.612 $\pm$0.006	0.447 $\pm$0.009	0.410 $\pm$0.011	0.414 $\pm$0.013	0.180 $\pm$0.010	0.456 $\pm$0.026	0.426 $\pm$0.007

The per benchmark variability is uneven and follows the size of the evaluation split. Standard deviations are smallest on the large single hop test sets and largest on Bamboogle, whose 125 examples make a single question worth 0.8 points. The gap between the two methods exceeds one standard deviation on every benchmark except Bamboogle, and the averages are separated by roughly two standard deviations. We describe the difference as consistent rather than as the outcome of a formal test, since three seeds is a small sample under reinforcement learning variance. Single benchmark comparisons on Bamboogle should be read with particular caution throughout this paper, including the comparison against GiGPO in the main results tables.

P CASE STUDY

Figure 9 follows one multi hop question through the schedule. It shows the rollout produced after the first exploration round, the privileged skeleton mined for this question after the second exploration round, and the rollout produced by the final policy. The example illustrates the behavior that the aggregate statistics of Appendix N describe, namely a change in the content of the queries at an unchanged number of retrievals.

Question. What is the date of death of Alexander of Masovia's father?
Gold answer. 21 January 1426

After $GRPO_1$, incorrect

<think> I need the date of death of Alexander of Masovia's father. </think>
<search> Alexander of Masovia father date of death </search>
<information> [Alexander of Masovia] Alexander of Masovia (1400 to 2 June 1444) was the second son of Siemowit IV, Duke of Masovia. </information>
<think> The passage gives the dates 1400 and 1444. </think>
<search> Alexander of Masovia death </search>
<information> [Alexander of Masovia] ... died on 2 June 1444 in Trento ... </information>
<answer> 2 June 1444 </answer>

Privileged skeleton mined after $GRPO_2$, teacher only

<search> Alexander of Masovia father </search> → <search> Siemowit IV Duke of Masovia date of death </search>

After $OPSD_3$, correct

<think> First identify the father, then look up his death date. </think>
<search> Alexander of Masovia father </search>
<information> [Alexander of Masovia] ... was the second son of Siemowit IV, Duke of Masovia. </information>
<think> The father is Siemowit IV. Now I need his date of death. </think>
<search> Siemowit IV Duke of Masovia date of death </search>
<information> [Siemowit IV] Siemowit IV (c. 1352 to 21 January 1426) was Duke of Masovia. </information>
<answer> 21 January 1426 </answer>

Figure 9: One 2Wiki question at three points of the schedule. The early policy collapses the two hops into a single query, retrieves a passage about Alexander himself, and reports his own dates. The privileged skeleton mined from a successful sibling rollout separates the two hops. The final policy issues the same number of retrievals as the early one but resolves the bridge entity first, which is the behavior the skeleton encodes.

The early rollout fails in a way that a trajectory level reward cannot diagnose. Both of its queries are well formed and both retrieve relevant passages, yet neither targets the father, so the chain never reaches the entity that the question actually asks about. The privileged skeleton contains no evidence and no answer, only the observation that a successful sibling resolved the bridge entity before querying for the date. Aligning the student to the teacher conditioned on this skeleton changes the distribution at the opening query positions, and the final policy reproduces the decomposition on questions of this shape. Note that the final rollout issues two retrievals, exactly as the failed one did, which is the individual instance of the aggregate finding that consolidation changes query content at an unchanged retrieval budget.

Q PROMPTS AND ALGORITHM DETAILS

Inference prompt. The student prompt follows the format used by prior search augmented reasoning work, so that the comparison against the outcome reward baselines is not confounded by prompt differences.

```
You are a helpful assistant. Answer the given question. You must
reason inside <think> and </think> first. If you lack knowledge,
you may call a search engine by writing a query inside <search>
and </search>, and the top passages will be returned inside
<information> and </information>. You may search as many times
as you need, up to four. When you know the answer, give it
inside <answer> and </answer> without any explanation.
Question: {question}
```

Figure 10: Standard inference prompt $P_S(q)$, used by the student and at evaluation time.

Privileged block. During consolidation the teacher receives one additional block, placed after the question and before the student rollout. It contains the search spans of the reference in order, joined by an arrow, and is truncated to 512 tokens. No outcome label is attached, since only successful references are used.

```
[Reference search plan] A previous successful attempt at this
question issued the following queries in order.
<search> {query 1} </search> -> <search> {query 2} </search> -> ...
```

Figure 11: Privileged block $\mathcal{M}(\tau_q^{\mathrm{ref}})$, visible to the teacher only.

Procedure. Figure 12 summarizes one complete cycle.

One Search-E1 cycle, starting from policy π_r and pool $\mathcal{D}_r$

1. Run GRPO on π_r for the scheduled number of steps using the outcome reward only, visiting a subset $\mathcal{V}_r \subseteq \mathcal{D}_r$, and obtain π_r^{RL}.
2. For every $q \in \mathcal{V}_r$, sample K rollouts from π_r^{RL} and record $c_r(q)$, the number of correct rollouts.
3. For every q with $c_r(q) > 0$, select the successful rollout with the fewest search calls, break ties by skeleton length, and store $\mathcal{M}(\tau_q^{\mathrm{ref}})$.
4. Set $\mathcal{D}_{r+1} = \mathcal{D}_r \setminus \{q \in \mathcal{V}_r : c_r(q) = K\}$.
5. Initialize the teacher and the student from π_r^{RL} and freeze the teacher.
6. For each consolidation step, sample τ^{stu} from the student under $P_S(q)$, evaluate the frozen teacher on τ^{stu} under $P_T(q, \tau_q^{\mathrm{ref}})$, and update the student on the clipped forward KL of Equation 9. Store a checkpoint every twenty steps.
7. Among stored checkpoints whose entropy retention on the calibration set is at least 0.85, select the one with the highest validation Exact Match and call it π_{r+1}.

Figure 12: One cycle of exploration and consolidation. No step introduces an external model, a reward model, or an annotation.